CLAIMS

What is claimed is:

1. A method for customizing the response for network based intrusion prevention comprising the steps of:

a) virtual proxying of the application data to enable custom response

b) enhancing transport layer to enable selective processing and selective modification of the steam for network based intrusion prevention

2. A method of claim 1, wherein component (a), virtual proxying, further comprising the steps of:

a) processing application data packets at packet level without terminating TCP or UDP sessions

b) using a processing engine to detect and prevent intrusions

c) using a custom interface with the transport layer to make any changes to application data or sessions

3. A method of claim 2, wherein component (c), using q custom interface with the transport layer, further comprising the steps of:

a) adding, removing, or modifying data in specified packets

b) adding or removing packets in the application packet stream

c) blocking or dropping application sessions

d) moving specified amount of existing or future data from packet controller or ingress queue to egress for direct transmission to destination

e) configuring access policing manager to specify what application streams can be forwarded at IP layer and what should be sent to vproxies

4. A method of claim 1, wherein component (b), enhancing transport layer, further comprising the steps of:

a) separating incoming packets in application flows using an access policy manager

b) enhancing TCP and UDP protocols to support virtual proxying

c) using a fault tolerant packet controller to send packets to virtual proxies and make necessary modifications to application data

5. A method of claim 4, wherein component (a), separating incoming packets in application flows, further comprising the steps of:

a) separating application packet flows that need to be forwarded at IP layer and send them to destination directly

b) optionally separating application packet flows that come from protected servers and sending them to state machine module for fast forwarding

c) separating application packet flows that need to go to vproxies for intrusion prevention processing and routing them to appropriate vproxies

6. A method of claim 4, wherein component (b), enhancing TCP or UDP protocol processing, further comprising the steps of:

a) adding ability to transmit and receive data using the same packets and perform all the protocol functions that are needed to maintain application sessions at the transport layer

2) enhancing session start procedure to automatically forward session start packets (for example SYN packets for TCP) and at the same time maintain their state in the local state machine and data structures

3) adding ability to add, remove, or modify data in the packets and take necessary action to make these operations seamless from source and destination perspective

4) selectively skipping the processing which is not needed.

7. A method of claim 4, wherein component (c), using a fault tolerant packet controller, further comprising the steps of:

a) providing a custom interface for vproxy to retrieve, pass, or modify application data packets and sessions

b) communicating with the packet controller of the standby packet controller to enable session level hot redundancy

c) performing tasks that enable automatic forwarding of vproxy configured selective data from ingress queue to egress queue

8. A method for customizing the processing for network or host based intrusion prevention comprising the steps of:

a) loading externally defined processing procedure libraries

b) combining multiple processing procedure libraries to form a unified processing engine that can be used for intrusion detection and prevention

c) removing old processing procedures that are not needed any more from the processing engine

d) adding new processing procedures in the processing engine to improve the intrusion detection and prevention

9. A method of claim 8 further comprising of:

a) building a semantic tree which contains trigger nodes and processing procedures

b) building the dynamic counterpart of trigger nodes as the application data traverse the semantic tree to detect and prevent intrusions

c) updating processing engine when old processing procedures are removed or new processing procedures are added

10. A method of claim 8, wherein component (b), combining multiple processing procedure libraries, further comprising of:

a) using init data structure functions from the library with the highest sequence number to create and initialize processing engine data structures

b) using init semantic tree functions from the library with the highest sequence number to create and initialize the processing engine semantic tree

c) using the populate semantic tree functions from all libraries to populate semantic tree with processing procedures

d) using full initialization functions from the library with the highest sequence number to initialize data structures

e) using incremental initialization functions from all libraries to initialize data structures

11. A method of claim 9, wherein component (c), updating processing engine, further comprising the steps of:

a) building the new semantic tree by loading the libraries for those processing procedures which should be part of the new processing engine

b) switching to new semantic tree for processing application data to detect and prevent intrusions